

Wright State University

CORE Scholar

---

Computer Science & Engineering Syllabi

College of Engineering & Computer Science

---

Fall 2006

## CS 780: Compiler Design and Construction I

Krishnaprasad Thirunarayan

*Wright State University - Main Campus*, [t.k.prasad@wright.edu](mailto:t.k.prasad@wright.edu)

Follow this and additional works at: [https://corescholar.libraries.wright.edu/cecs\\_syllabi](https://corescholar.libraries.wright.edu/cecs_syllabi)



Part of the [Computer Engineering Commons](#), and the [Computer Sciences Commons](#)

---

### Repository Citation

Thirunarayan, K. (2006). CS 780: Compiler Design and Construction I. .

[https://corescholar.libraries.wright.edu/cecs\\_syllabi/196](https://corescholar.libraries.wright.edu/cecs_syllabi/196)

This Syllabus is brought to you for free and open access by the College of Engineering & Computer Science at CORE Scholar. It has been accepted for inclusion in Computer Science & Engineering Syllabi by an authorized administrator of CORE Scholar. For more information, please contact [library-corescholar@wright.edu](mailto:library-corescholar@wright.edu).

# CS 780 Compiler Design and Construction I

---

- **Instructor** : T. K. Prasad
  - **Phone No.** : (937)-775-5109
  - **Email** : [t.k.prasad@wright.edu](mailto:t.k.prasad@wright.edu)
  - **Home page**: <http://www.cs.wright.edu/~tkprasad/>
  - **Quarter** : Fall, 2006
  - **Class Hrs** : TTh, 6:05pm-7:20pm, 012 Library Annex
  - **Office Hrs** : TTh, 3:30pm-4pm, 5:30pm-6pm, 337 RC (or by appt.)
- 

## Table of Contents

1. [Course Description](#)
  2. [Prerequisites](#)
  3. [Texts and References](#)
  4. [Relevant Websites](#)
  5. [Course Load and Grading](#)
  6. [Class Schedule, Syllabus, and Lectures](#)
  7. [Assignments and Exams](#)
  8. [Late Submission Policy](#)
  9. [Collaboration vs Academic Misconduct](#)
  10. [Research Journals and Conferences](#)
- 

## Course Description

This course deals with the theory and practice of compiler design. Topics emphasized are scanning and parsing. If time permits, semantic analysis will also be covered.

---

## Prerequisites

- [Formal Languages and Automata](#) (CS466/CS666)
  - [Comparative Languages](#) (CS480/CS680)
- 

## Required Texts

1. *Compiler Construction: Principles and Practice*, Kenneth C. Loudon, PWS Publishing Company,

1997.

2. *lex & yacc, 2nd edition*, John R. Levine, Tony Mason & Doug Brown, O'Reilly & Associates, Inc, 1992.

## References

1. *Modern Compiler Implementation in Java (2nd Ed.)*, Andrew Appel, Cambridge University Press, 2002.
2. *Compilers: Principles, Techniques, and Tools*, Aho, Sethi and Ullman, Addison Wesley, 1988.
3. *The C++ Programming Language, Third Edition*, Bjarne Stroustrup, Addison Wesley, 1997.

## Relevant Websites

- ONLINE HELP
  - GNU Manuals  
<http://www.delorie.com/gnu/docs/>

### PROGRAMMING LANGUAGES AND COMPILERS ON THE WEB

- The Compiler Connection  
<http://www.compilerconnection.com/>
- Compiler News Group  
[news:comp.compilers](mailto:news:comp.compilers)

## Course Load

The course load includes three programming assignments based on the COOL compiler project worth 35 points, a midterm worth 30 points and a final worth 35 points. Normally, exams are open book and open lecture notes.

## Grading

The letter grades will be assigned using the following scale: A[90-100], B[80-90), C[70-80), D[60-70), and F[0-60). However, I reserve the right to adjust the scale somewhat to utilize the gaps in the distribution. Academic dishonesty will be "rewarded" with a grade of "F". "Sharing/reuse" of solutions to assignment problems is strictly prohibited.

## Class Schedule, Syllabus and Lectures

	Topic	Readings
Class 1	<u>Introduction to Compilers</u>	Chapter 1
Class 2	<u>Introduction to COOL</u>	CoolAid

Class 3	<u>Lexical Analysis</u>	Chapter 2
Class 4	<u>Implementing Lexer</u>	
Class 5	<u>FLEX</u>	Chapters 1-2, 6 lex & yacc
Class 6	<u>Context-free Grammars</u>	Chapter 3
Class 7	<u>Ambiguity; Abstract Syntax Trees</u>	
Class 8	<u>BISON</u>	
Class 9	<b>Midterm Exam (Oct 10)</b>	
Class 10	<u>Top-down Parsing</u>	Chapter 4
Class 11	(continue)	
Class 12	<u>Bottom-up Parsing</u>	Chapter 5
Class 13	<u>Bottom-up Parsing : Basics (ppt)</u>	
Class 14	<u>Bottom-up Parsing : Algorithms</u>	
Class 15	(continue)	
Class 16	<u>LR Parsing</u>	
Class 17	(continue) (ppt)	
Class 18	<u>Overview of Semantic Analysis</u>	Chapter 6
Class 19	Type Checking and Inference	
Class 20	(WRAP-UP)	
<b>Final Exam (Nov 16) (8pm-10pm)</b>		

---

---

### ASSIGNMENTS (Fall 2006)

- Cool Compiler Project - Programming Assignment 1 ([pdf](#))
  - Cool Compiler Project - Programming Assignment 2 ([pdf](#))
  - Cool Compiler Project - Programming Assignment 3 ([pdf](#))
- 

### RELATED DOCUMENTS

- CoolAid Manual (Alex Aiken) ([ps](#)) ([pdf](#))
  - A Tour of the Cool Support Code (Alex Aiken) ([ps](#))([pdf](#))
- 
- 

### EXAMINATIONS (Fall 2005)

- Midterm ([pdf](#))

- Final ([pdf](#))

---

### **Late Submission Policy**

You will lose 5% for each day an assignment is submitted late; assignments will be accepted up to 4 days past the submission time. Late penalty is accrued on weekends just as during the week. For every 24 hour period (or fraction thereof) that the assignment is late, 5% is deducted from your score. Partial credit will be given to students who turn in partially completed assignments. Electronic submission will be used for all assignments.

Special considerations will be given for students who have a medical excuse for late submission (written notification from a doctor is required). These considerations may extend to medical emergencies involving children or other family members. Such consideration is at the discretion of the instructor. Special consideration may also be given for employment conflicts (e.g. military duty, travel) if brought to the attention of the instructor prior to the due date for an assignment.

Course requirements for other courses are NOT a valid reason for special consideration.

---

### **Collaboration vs Academic Misconduct**

You are encouraged to exchange ideas regarding your programming assignments with your classmates. However, you must turn in your own work for each assignment (unless I explicitly assign a group project). As such, you should discuss programming assignments at the conceptual level only and should not share your code with your classmates. You should also never recycle printouts in public computer labs, never leave your workstation unattended without first locking the screen, always wait until your printouts have printed or delete them from the print queue when using public printers, always remove any files that you have downloaded to the PC in public PC labs, etc.

Submitting a program as your own, when some or all was written by someone else is an act of plagiarism and constitutes academic misconduct. The minimum penalty for such a misconduct is a score of 0 on the assignment. See your student handbook for more information on academic misconduct and its consequences. Misconduct will be handled in accordance with university policy.

---

### **LINKS TO RESEARCH JOURNALS AND CONFERENCE PROCEEDINGS**

- ACM Transactions on Programming Languages and Systems  
<http://www.cs.wustl.edu/~toplas/>
- ACM SIGPLAN sponsored conferences  
<http://www.acm.org/sigplan/#conferences>
- Upcoming Compiler and Parallel Computing Conferences  
<http://www.cs.rice.edu/~roth/conferences.html>

---

I. K. Prasad (08/21/06 03:13:04 PM )